# Fonts

* Fonts/google.com
  + Josefine Sans
    - Thin 100, Extralight 200, Light 300, Medium 500, Semibold 600, Bold 700
  + Lora
    - Regular 400 italic, Medium 500 italic, SemiBold 600 italic
  + Varela Round
  + Varela Regular 400
* Font Awesome – icons
  + https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css

# JavaScript Naming Standards

* 1. Variables camel Case
  2. Booleans – prefix with “is” or “has” then camel case. Ex: hasOwner
  3. Functions – camel case and use nouns and verbs. Ex: getName.
  4. Constants – UPPER\_SNAKE\_CASE. All upper case. Ex: DAYS\_TO\_GO.
  5. Classes – pascal case. Ex: DogCartoon.
  6. Components – pascal case. Ex: CatPerson.
  7. Methods – camel case. Ex: storeRecord.
  8. Private Function – prefix with underscore and camel case. Ex: \_toonName.
  9. Global Variables – camel case for mutable variables and uppercase for immutable variables.
  10. File Names – lower case.

# Project Steps

1. $ npm init -y
2. $ npm i express dotenv mongoose bcryptjs jsonwebtoken cookie-parser
3. $ npm -D nodemon
4. Update package.json with “type”: “module”.
   1. Use ES6 modules instead of CommonJS by adding an entry into the package.json file.  Add the following: "type": "module". This allows you to use the import function.
5. Add a script in the package.json file.
   1. “start”: “node backend/server.js”
   2. “server”: “nodemone backend server.js”
6. Implement dotenv
   1. Create the .env file under the root folder.
      1. NODE\_ENV=development
      2. PORT=8000
7. Add gitignore file
   1. Create the .gitignore file under the root folder and add the following
      1. node-modules
      2. .env
8. Create the ‘routes’ folder under the backend folder
9. Create a file named ‘userRoutes.js’ under the ‘routes’ folder.
10. Create a folder under the ‘backend’ folder named ‘controllers’.
    1. These controller methods get the request from the routes and convert them to HTTP responses with the use of any middleware as necessary.
11. Create a file named ‘userController.js’ under the ‘controllers’ folder.
12. $ npm i express-async-handler
    1. Used to manage async processing.

PostMan

1. Create a personal workspace named: JWT Auth
2. Create a new environment to define the baseURL path.
3. Create a new Collection to store the HTTP method queries.